Electric Piano

EE 45 Final Project

Garrett Knuf and Deepro Pasha

California Institute of Technology
June 2023

Contents

1 Introduction

An electric piano synthesizer was engineered utilizing many of the technologies and analysis techniques demonstrated in EE 45, Electronic Systems and Laboratory. The piano can play 12 different notes ranging from C4 to B4, spanning an entire octave. All 12 keys of the piano generate frequencies close to the exact frequencies of the notes with minimal error. A custom GUI runs on a computer and interfaces with the system to control the frequency, volume, and tone of the note being played. The system is responsible for generating a waveform from user input, filtering the waveform, modifying the amplitude of the waveform, and then outputting the waveform to a speaker.

2 System Architecture

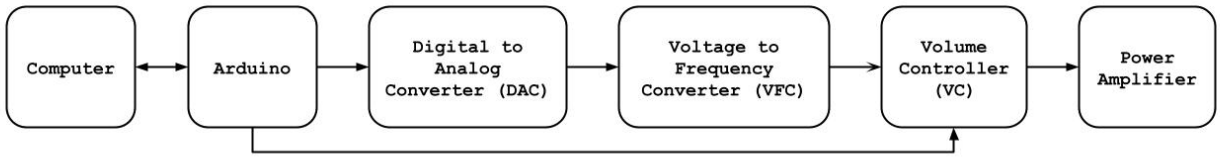

Figure 1: Top level block-diagram of system

Figure 1 depicts the modularity of the system. User input is interpreted by the Graphical User Interface (GUI) running on the computer and communicates it to the Arduino via USB. The Arduino sets its digital output pins accordingly so that the Digital to Analog Converter (DAC) can output an analog voltage that varies depending on which note should be played. The Voltage to Frequency Converter (VFC) receives the output of DAC as an input and converts the analog voltage to a waveform of the desired frequency. The Volume Controller (VC) then adjusts the amplitude of the waveform as determined by a signal from the Arduino. This waveform is then output to a power amplifier which plays the note through a speaker.

2.1 Digital to Analog Converter (DAC)

Since there are the 12 notes on the chromatic scale the piano plays, there are 12 inputs from the Arduino to the DAC, and each input corresponds to a different note. For each note, a different digital pin (B0 to B11) will be set to digital output HIGH (about 5 V) while all other pins will be set to high impedance mode (Hi-Z). Setting the non-relevant pins to HI-Z essentially disconnects them from the circuit and this can be accomplished by setting these Arduino pins to digital input mode instead of digital output mode. Figure 2 shows the connections between the Arduino and the DAC.

The DAC internally consists of a voltage divider connected to all the Arduino pins. It varies in voltage as the Arduino pins change, and this voltage connects to a buffer to ensure the voltage divider does not vary as current is drawn from the output of the DAC. Figure 3 shows the internal workings of the DAC.

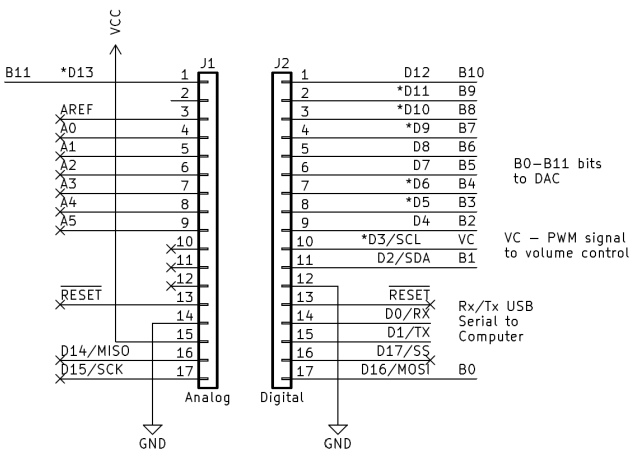

Figure 2: Arduino pin-out to DAC and volume control module

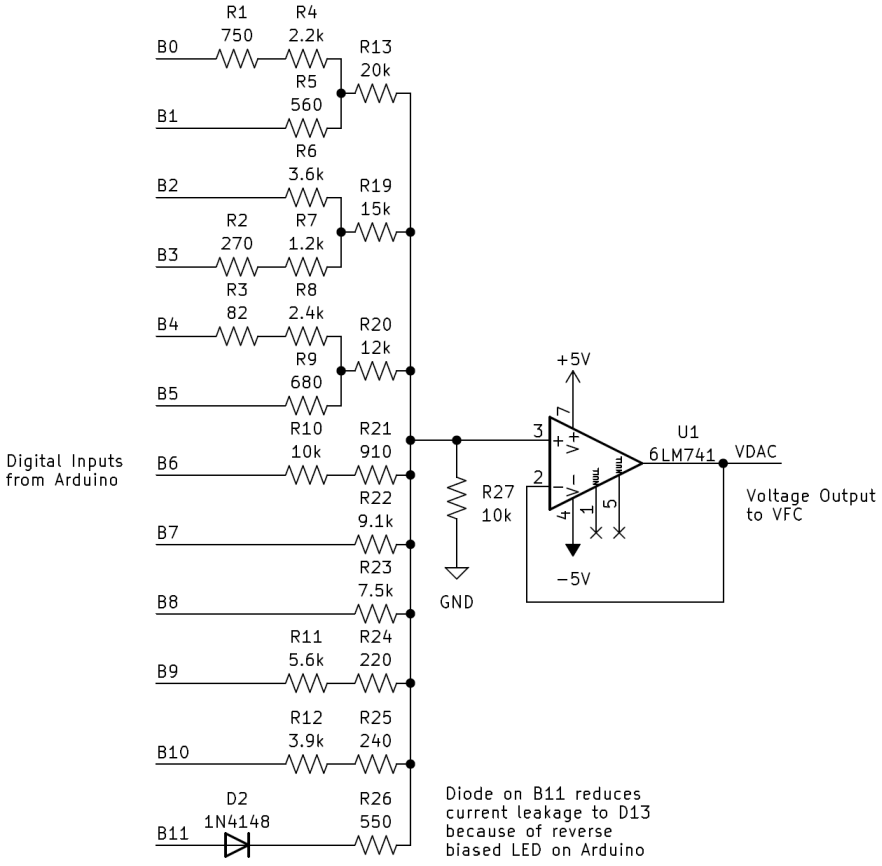

Figure 3: Digital to Analog Converter schematic

2.2 Voltage to Frequency Converter (VFC)

The VFC converts analog voltages output by the DAC into waveforms with corresponding frequencies. Before any filtering occurs, the VFC generates a square wave (Figure 4). The components values of the circuit are calibrated so that op amp U3 operates as a Schmitt trigger, triggering between saturation values as the capacitor charges and discharges periodically.

The square wave is then passed into a three stage low pass filter (Figure 5). This works to remove harsh frequencies and harmonics that may distort the output waveform, giving the final output a cleaner tone. The original square wave becomes smoother and starts to resemble a sine wave as it gets filtered.

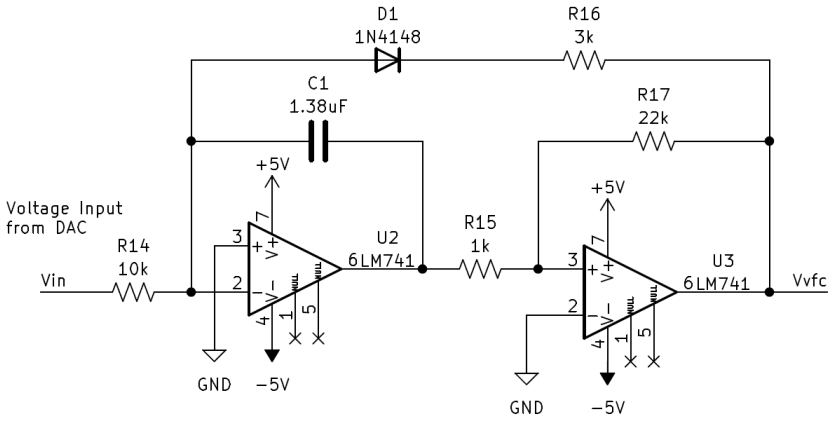

Figure 4: Voltage to Frequency Converter schematic

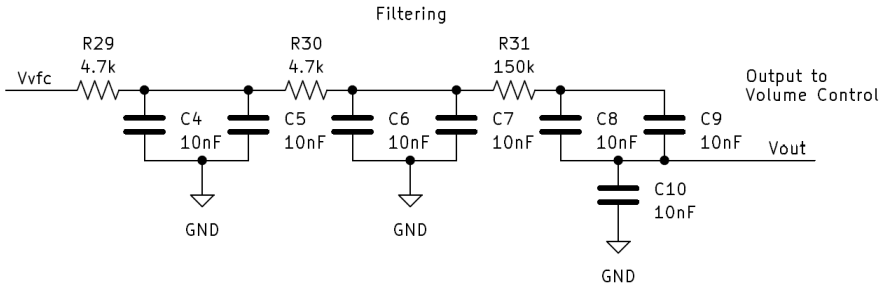

Figure 5: Three stage low pass filter schematic

2.3 Volume Controller (VC)

The volume controller works by varying the drain voltage of a common source stage transistor being operated in the triode region. Since the Arduino is not capable of producing an analog voltage, one its digital pins is used to create a PWM signal (Figure 6). However, since the piano is a frequency sensitive application with potential cutoff frequencies, it was necessary to convert this to an analog signal. The Arduino can control the duty cycle of this signal, so a PWM to analog voltage converter was added before this voltage is passed into the resistive load on the drain of the transistor. Because the gain of the transistor stage varies with drain voltage, the amplitude of the waveform is affected as well, thus adjusting the volume of the system.

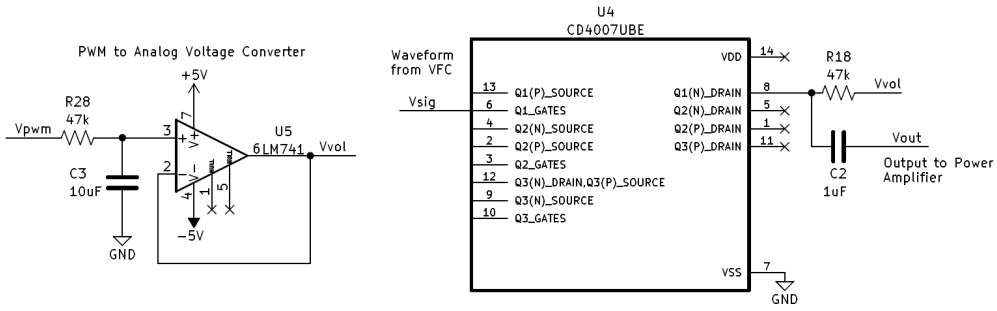

Figure 6: Volume controller schematic

3 Theoretical Analysis

A theoretical analysis of the electric piano modules can provide a clearer understanding of how the component values contribute to the functionality of the system.

3.1 Digital to Analog Converter (DAC)

To understand the operation of the DAC, a generic version of the circuit for all possible notes can be created (Figure 7). V_{in} depicts an input voltage from an Arduino pin set to digital HIGH, R_1 is a pull-down resistor with a fixed value, and R_2 varies depending on which pin of the Arduino is connected. V_{out} is the output voltage, and because it may draw current, a buffer is used so the output does not sink the voltage across the voltage divider. Thus, the transfer function of the DAC can be determined.

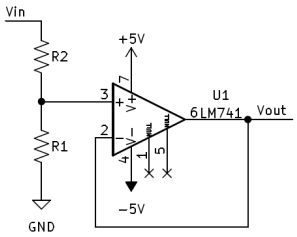

Figure 7: Generic schematic to model DAC

$$\frac{V_{out}}{V_{in}} = \frac{R_1}{R_1 + R_2}$$

It is clear that by properly varying R_2 between different pins of the Arduino, the output voltage can be any value between 0 V to 5V (the maximum output voltage of Arduino digital pins).

3.2 Voltage to Frequency Converter (VFC)

The output voltage from the DAC becomes the input voltage of the VFC. To understand the operation of the VFC, a generic circuit for all possible component values can be developed.

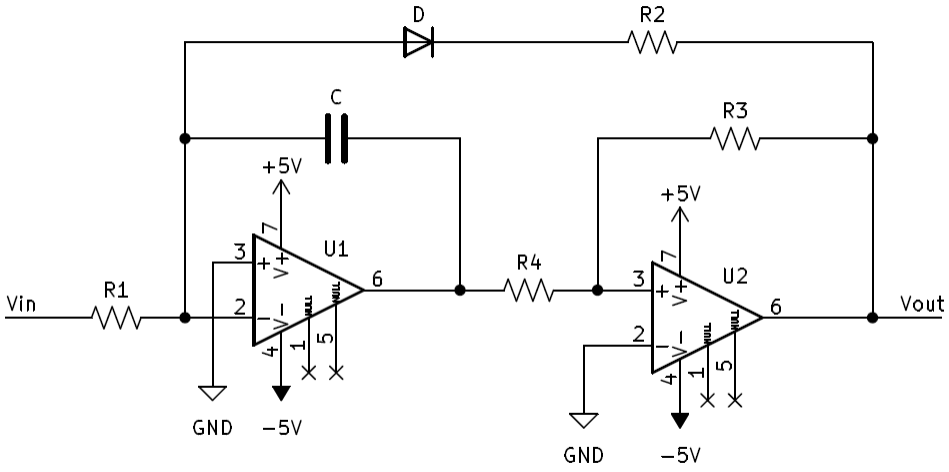

Figure 8: Generic VFC circuit

The op amp on the left behaves like an integrator. This causes its output voltage to accumulate with a sign opposite to that of the input voltage V_{in}. The op amp on the right behaves like a non-inverting Schmitt trigger due to the positive feedback it receives. This indicates that a lower and upper threshold voltage exists for the Schmitt trigger to output a positive or negative saturation voltage.

The region of the circuit that operates as a Schmitt trigger with R_3, R_4, and U_2 will be analyzed first. The voltage on the node shared between C, R_4, and U_1 will henceforth be defined as V_1. To find the lower threshold voltage of the Schmitt trigger, it is assumed $V_{OUT} = V_{SAT}$ (where V_{SAT} is the saturation voltage of the op amp) since this is true the moment before the lower threshold stage will make the output voltage become the negative saturation voltage. It is also assumed here that V_1 is decreasing from its upper threshold, causing $V+$ to approach 0 from the negative side. $V+$ will be negative at the upper threshold in order to stop the op-amp from giving higher voltages than V_{SAT}. $V+$ thus decreases in magnitude as V_1 decreases from the upper threshold as the amount over-delivered by the op-amp decreases in a proportional manner to V_1's reduction. With this, the lower threshold voltage of the Schmitt trigger can be determined.

$$V_L = -\frac{R_3}{R_4} V_{SAT}$$

To find the upper threshold voltage, it is assumed $V_{OUT} = -V_{SAT}$ since this is the moment before the upper threshold stage makes the output voltage become the positive saturation voltage. $V+$ also approaches 0 from the positive side from the op amp, providing voltage to ensure that the output voltage is kept at V_{SAT} for the stage of the upper threshold. With this, the upper threshold voltage of the Schmitt trigger can be determined.

$$V_U = \frac{R_3}{R_4} V_{SAT}$$

When the voltage of the node shared between C and R_4 (which is now defined as V_1) becomes V_U, the output voltage becomes V_{SAT}, and when that same node has a voltage of V_L, the voltage become $-V_{SAT}$. When

the latter situation occurs, the diode on the top branch will short the path, letting current flow through. The node between C and R_1 can then be analyzed to gain understanding into the variance of V_1 when the output voltage is $-V_{SAT}$. That node will be set to 0 V due to the negative feedback of the ideal op-amp and the following circuit analysis equations are developed:

$$\frac{V_{IN}}{R_1} = \frac{V_{SAT}}{R_2} - C\frac{dV_1}{dt}$$

$$\frac{dV_1}{dt} = \frac{1}{C}(\frac{V_{SAT}}{R_2} - \frac{V_{IN}}{R1}).$$

Solving the differential equation:

$$V_{1@t} - V_{1@t_0} = \frac{1}{C}(\frac{V_{SAT}}{R_1} - \frac{V_{IN}}{R1})t - t_0$$

.

Let t_0 be 0 and $V_{1@t_0}$ be the lower threshold.

$$V_1 = -\frac{R_3}{R_4}V_{SAT} + \frac{1}{C}(\frac{V_{SAT}}{R_2} - \frac{V_{IN}}{R1})t.$$

Thus, V_1 behaves in a linear manner with respect to time. Note that the period of the V_1 waveform is the time it takes for V_1 to traverse from the lower to upper to lower threshold. The period of repetition for V_{OUT} will match the period of repetition for V_1 as shown:

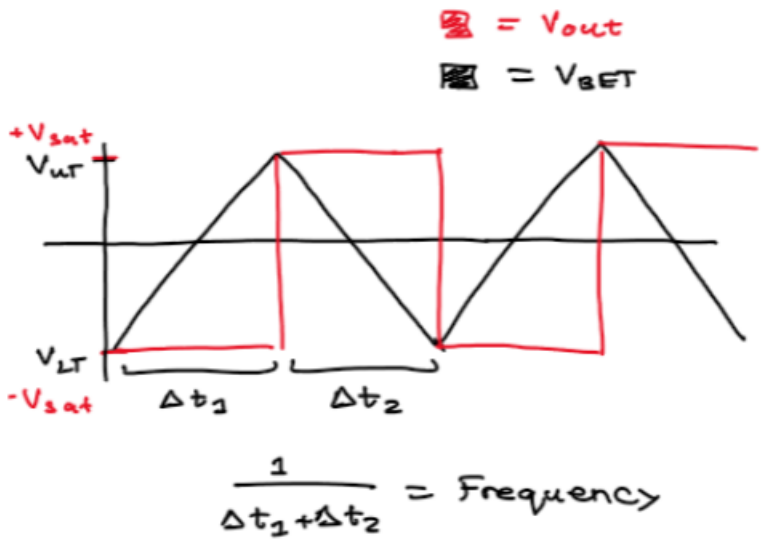

Figure 9: Matching of period between V_{OUT} and V_1

The frequency of V_{OUT} as a waveform can thus be found from this period. Looking to find this period, V_1 increases linearly from the lower to upper threshold when the path is opened for the diode. This time it takes to increase can be found as so:

$$V_1 = -\frac{R_3}{R_4}V_{SAT} + \frac{1}{C}(\frac{V_{SAT}}{R_2} - \frac{V_{IN}}{R1})t$$

$$\frac{R_3}{R_4}V_{SAT} = -\frac{R_3}{R_4}V_{SAT} + \frac{1}{C}(\frac{V_{SAT}}{R_2} - \frac{V_{IN}}{R1})t$$

$$2C\frac{R_3}{R_4}V_{SAT} = \frac{1}{C}(\frac{V_{SAT}}{R_2} - \frac{V_{IN}}{R1})t$$

$$t = \frac{2CR_3V_{SAT}}{R_4}(\frac{R_2R_1}{V_{SAT}R_1 - V_{IN}R_2}).$$

Searching for the time it takes for V_1 to go from the upper to lower threshold, it should be examined where $V_{OUT} = V_{SAT}$. Examining the capacitor node but with a closed diode path:

$$\frac{V_{IN}}{R_1} = -C\frac{dV_1}{dt}$$

Solving this differential equation, the following result is acquired

$$V_{1@t} - V_{1@t_0} = \frac{V_{IN}}{R_1C}(t - t_0)$$

Let t_0 be 0 and $V_{1@t_0}$ be the upper threshold threshold.

$$t = \frac{2R_3R_1CV_{SAT}}{R_4V_{IN}}$$

The frequency of V_{OUT} can therefore be found:

$$f = \frac{1}{\frac{2CR_3V_{SAT}R_2R_1}{R_4(V_{SAT}R_1 - V_{IN}R_2)} + \frac{2CR_2V_{SAT}R_1}{R_4V_{IN}}}$$

$$f = \frac{V_{IN}R_4}{2CV_{SAT}R_1R_3} - \frac{V_{IN}^2R_2R_4}{2CV_{SAT}^2R_1^2R_3}.$$

A low pass filter was added to the output of the VFC to convert its square wave output to a sine wave output. The low pass filter was then implemented from a design similar to https://circuitdigest.com/electronic-circuits/simple-square-wave-to-sine-wave-converter. This is the generic design of the low pass filter:

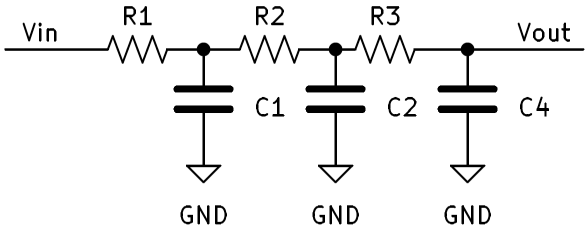

Figure 10: Generic three stage low pass filter design

The low pass filter is composed of three low pass filters put in series. This first low pass filter works to first form a parabola shape for the rise and fall of the square wave via the discharge and charge of the capacitor. The second low pass filter next works to take that parabola shape and use the same discharge and charge

capability to until the parabola slopes into straight slopes, forming a triangle wave. Finally, the third low pass filter takes in the triangle wave as an input and using again the same discharge and charge capability of the RC circuit turns that into a sine wave. That thus gives us a sine wave from the square wave. Exact values for resistors and capacitors were determined in lab.

3.3 Volume Controller (VC)

The first stage of the volume controller converts the PWM signal of the Arduino to an analog voltage. This can be accomplished with a low pass RC filter to isolate the DC component of the signal. The Arduino has the capability to vary the duty cycle of the signal, so a larger duty cycle correlates to a higher analog voltage that can vary between 0 V and the 5 V maximum output voltage of the digital pins. The filter can be optimized by making as large of an RC time constant as possible. A lower cutoff frequency also correlates to less signal noise. A buffer is also used on the output to stabilize the signal so it remains unaffected from any current draw. The following cutoff frequency should be made as small as possible.

$$f_c = \frac{1}{2\pi RC}$$

An NMOS transistor was utilized here for the voltage controller. It was implemented as a common source amplifier as such https://left.engr.usu.edu/courses/.

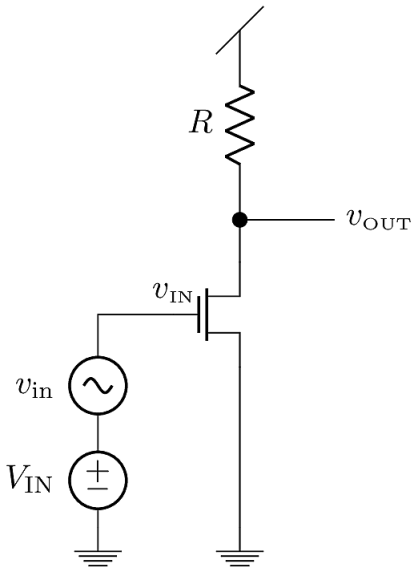

Figure 11: Volume controller transistor design

Based off the small signal model, it is found that $A_V = -g_m R_D$. It is desired to control this gain via control of the g_m. The best way to do this was to put the NMOS transistor into triode region and control the drain voltage. The gain is non-constant in triode regiode since g_m varies as demonstrated through the following graph in the textbook Sandra and Shea.

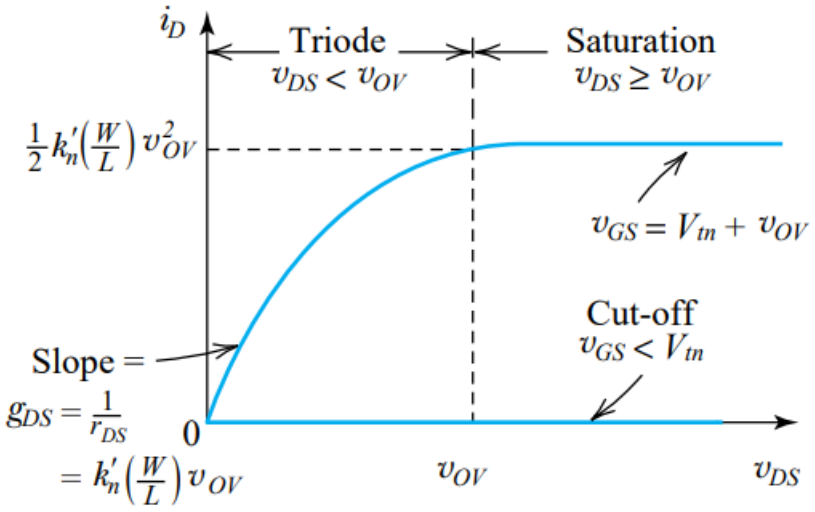

Figure 12: Triode Curve

It was assumed that the gain would follow similar behavior to that of the small signal model in saturation mode when the NMOS transistor was also in triode region. As can be seen, selecting different values of V_{DS}, which is equal to V_{DD}, would change the slope of the i_D to V_{DS} curve, thus changing the g_m value accordingly. Thus, varying between 0 and some set value of V_{DS} in some select time interval to shifts between some value of gain and 0 gain, thus forming a decay between a waveform with some volume and no volume at all. This thus created the volume decay effect desired.

4 Simulations

To verify decision choices and determine appropriate component values for each module of the system, simulations were carried out in LTSpice.

4.1 Digital to Analog Converter (DAC)

Figure 13 demonstrates that varying R_2 provides a suitable range of voltages for the DAC to operate. It shows how increasing R_2 decreases the output voltage, so the DAC will be properly functional if suitable choices are made for resistor values connected to Arduino pins.

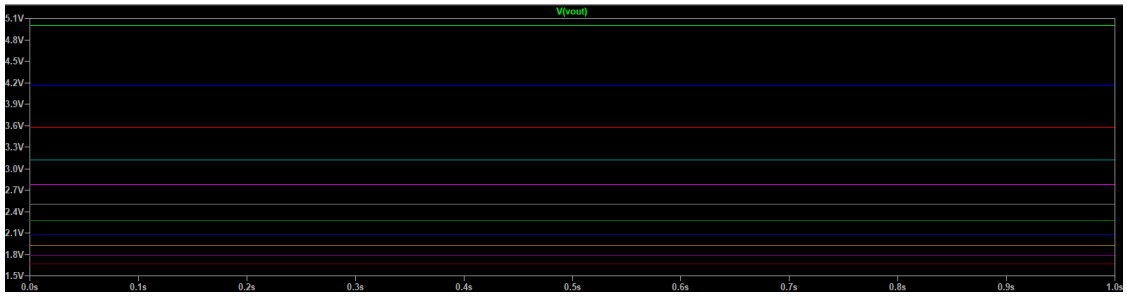

Figure 13: DAC simulation with a parameter sweep on the resistor R_2 ranging from 1 ohm to 20 kohm with steps of 2 kohm. Notice that DAC voltages vary from about 1.6 - 4.9 V.

4.2 Voltage to Frequency Converter (VFC)

A successful simulation of the VFC requires that it can support the entire range of frequencies needed to play all notes of the piano (about 250 - 500 Hz). The exact component values can be determined when testing the circuit, however, the simulation needs to reflect how its linearity and duty cycle are affected as the input voltage changes. Observe the three simulations completed below. Figures 14, 15, and 16 clearly exhibit the dependency that frequency has on the input voltage. Though the duty cycle is notably above 50%, it spans all desired frequencies with close to linear behavior. So, this design satisfies the requirements.

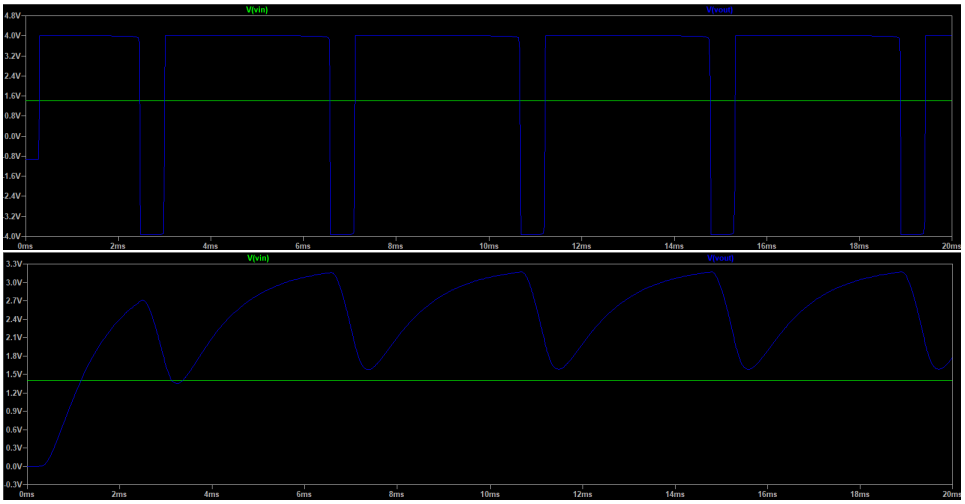

Figure 14: Comparison of VFC unfiltered (top) and filtered (bottom) with $V_{in} = 1.4V$ and $f = 244$Hz

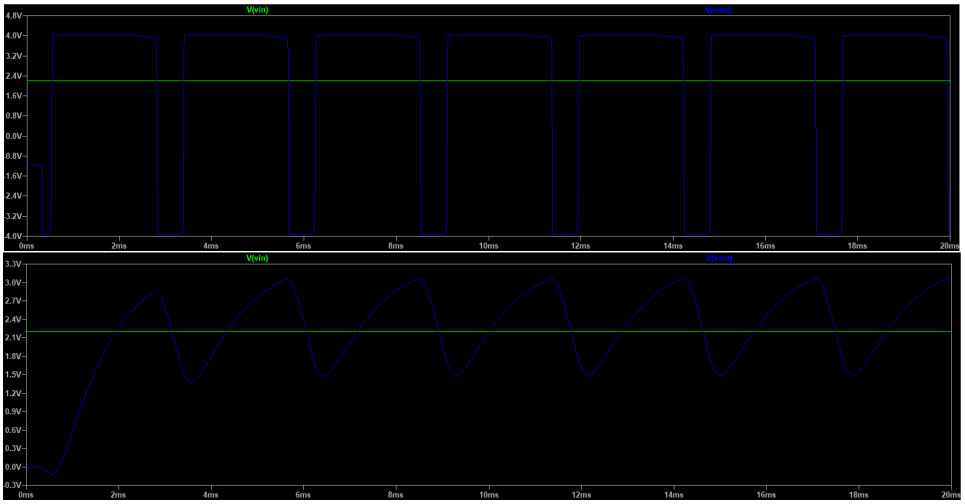

Figure 15: Comparison of VFC unfiltered (top) and filtered (bottom) with $V_{in} = 2.2V$ and $f = 350$Hz

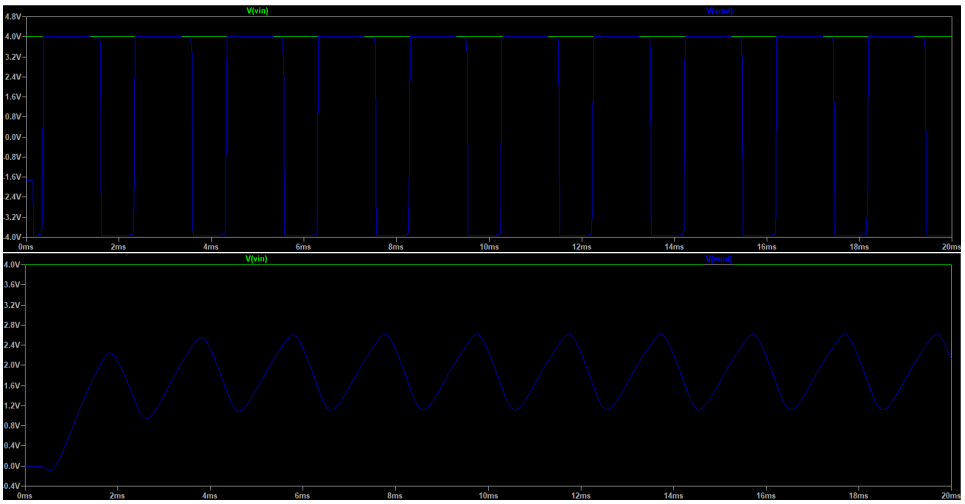

Figure 16: Comparison of VFC unfiltered (top) and filtered (bottom) with $V_{in} = 4.0V$ and $f = 505$Hz

4.3 Volume Controller (VC)

It was difficult to simulate the volume controller in LTSpice when trying to simulate the transistor in triode region. Some attempts were made, but ultimately, were not successful. It was thought best the implement the circuit through trial and error in the laboratory to find component values since there was already a proper theoretical understanding about how the circuit operates.

5 Implementation

Utilizing the models developed in Section 3 and the data collected in Section 4, the modules were assembled. The components were previously assumed to be ideal, however, upon testing it was clear that real-world components are non-ideal. So, components values were adjusted where necessary to compensate.

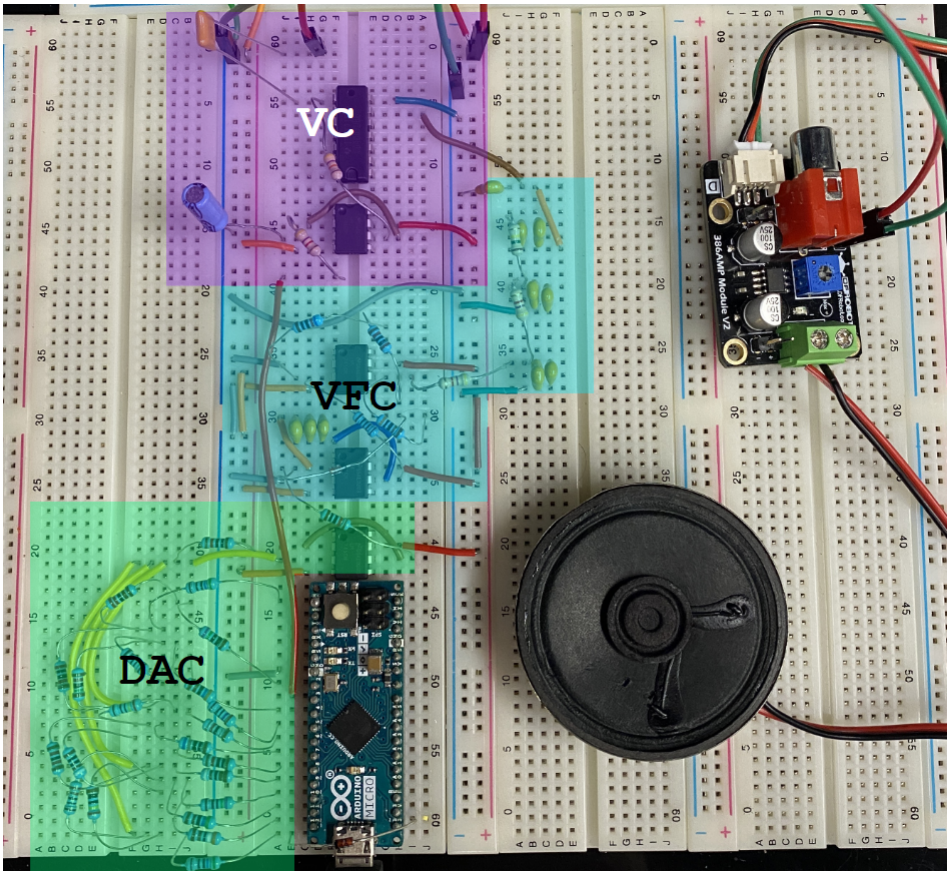

Figure 17: Completed circuit on breadboard

5.1 Digital to Analog Converter (DAC)

The digital pins of the Arduino are supposed to output 5 V, but upon measuring them it was found they were much closer to 4.92 V, so this value was used going forward. It was also discovered that using Arduino pin 13 was not working as expected when it was set to high impedance mode since it was drawing current. Because the Arduino board contains an LED on pin 13, it can become reverse biased in high impedance mode, drawing current and sinking the voltage of the voltage divider. To account for this, a diode was connected to prevent current flowing into the Arduino on pin 13, and it assumed that there was a constant 0.7 V drop across it when doing calculations.

The required output voltages of the DAC were determined by the VFC input voltages determined in Table 2. It was decided that $R_1 = 10k\Omega$, so referring to Figure 7, the required resistance values of the DAC can be calculated.

$$R_2 = \frac{R_1(V_{in} - V_{out})}{V_{out}} = \frac{10^4(4.92 - V_{out})}{V_{out}}$$

Note	Frequency (Hz)	DAC Output (V)	Calculated R_2 (Ohms)	Actual R_2 Used (Ohms)
C	261.63	1.50	22800.00	22750
C#	277.18	1.61	20559.01	20560
D	293.66	1.72	18604.65	18600
D#	311.13	1.86	16451.61	16470
E	329.63	2.01	14477.61	14482
F	349.23	2.17	12672.81	12680
F#	369.99	2.35	10936.17	10910
G	392.00	2.57	9143.97	9100
G#	415.30	2.81	7508.90	7500
A	440.00	3.11	5819.94	5820
A#	466.16	3.48	4137.93	4140
B	493.88	4.00	550.00	560

Table 1: The resistor values implemented in the circuit are shown here. Based on the availability of components available in the lab, the values were slightly modified. Multiple resistance values were achieved by connecting resistors in series as shown in Figure 3

.

5.2 Voltage to Frequency Converter (VFC)

The biggest challenge of designing the VFC was managing the trade-off between linearity and duty cycle. Linearity of the circuit helps with tuning the frequencies of each note. Since the DAC provides a certain resolution (meaning it may have slight error), having the voltages that are needed for each note farther apart can reduce errors in frequency. Having a near 50% duty cycle is also optimal for producing the best sound quality. However, when designing the circuit a balance was found where the circuit was almost linear with a 60% to 85% cycle. It was determined that prioritizing linearity was slightly more important than duty cycle since the tone quality is irrelevant if the piano is out of pitch. The VFC component values are shown in Figure 4 and the process for determining them heavily consisted of running simulations in LTSpice as shown in Section 4.2. Using LTSpice, the input voltages required to output were each frequency were also determined.

Note	Frequency (Hz)	VFC Input (V)
C	261.63	1.50
C#	277.18	1.61
D	293.66	1.72
D#	311.13	1.86
E	329.63	2.01
F	349.23	2.17
F#	369.99	2.35
G	392.00	2.57
G#	415.30	2.81
A	440.00	3.11
A#	466.16	3.48
B	493.88	4.00

Table 2: Input voltages required by the VFC to output a waveform with frequencies corresponding to the piano notes.

The low pass filter on the output of the VFC was implemented as demonstrated by Figure 5. The resistor and capacitor values were chosen to maximize the sinusoidal nature of the filtered waveform. The process to find these component values was primarily experimenting with different values and observing how they affect

the waveform on the oscilloscope. It was also important to ensure that too much filtration did not occur to prevent higher frequencies (around 400-500 Hz) from being inaudible when played through the speaker.

5.3 Volume Controller (VC)

The volume controller was implemented as demonstrated by Figure 6. Resistor and capacitor values were selected as to prevent the cutoff frequency of the transistor from being too high. This allowed all of the notes and their frequencies to be played. In addition, a buffer was added to provide feedback for a more stable signal. A large RC constant was made using a 10uF capacitor, and the resistor value would be determined with trial and error during testing. This would work to turn the PWM of the Arduino, which had a duty cycle when raised to a voltage value, into a singular voltage value with no duty cycle.

The CD4007 transistor was used and implemented in the manner described in the theoretical section - as a common source amplifier. R_D was selected to ensure that the gain received from the transistor implementation was not too low as a result of the triode region while also balancing it with the cutoff frequency of the transistor. An additional capacitor was also implemented on the output to act as a coupling capacitor, thus ensuring that there was no DC offset. This allowed us to maximize the amount of waveform coming out of the transistor in a stable manner.

6 Results/Plots/Codes

To ensure reliability of the system, each module was tested. Data was collected and examined to determine the overall effectiveness concerning the performance of the electric piano.

6.1 Digital to Analog Converter (DAC)

Note	Ideal Voltage (V)	Measured Voltage (V)	Resistance	Error (%)
C	1.50	1.53	22.75k	1.807
C#	1.61	1.65	20.56k	2.317
D	1.72	1.76	18.6k	2.308
D#	1.86	1.90	16.47k	2.124
E	2.01	2.08	14.482k	3.348
F	2.17	2.24	12.68k	3.281
F#	2.35	2.41	10.91k	2.651
G	2.57	2.65	9.1k	2.934
G#	2.81	2.89	7.5k	2.790
A	3.11	3.18	5.82k	2.383
A#	3.48	3.58	4.14k	2.813
B	4.00	4.15	0.56k	3.638

Table 3: Data collected from the assembled and tested DAC

As shown by Table 3 there is an average DAC voltage error of less than 3%. This is a reasonable error that should not be problematic unless the frequencies measured in Section 6.2 have significant error.

6.2 Voltage to Frequency Converter (VFC)

Table 4 shows the difference between the desired frequency (Hz) and the measured frequency (Hz). As can be seen, the frequencies measured are close to the frequencies desired with minimal error (averaging less than 1 Hz error). Thus, the VFC generates frequencies that are very close in sound to the tones desired that are close enough to be audibly indistinguishable from the actual note frequency to most.

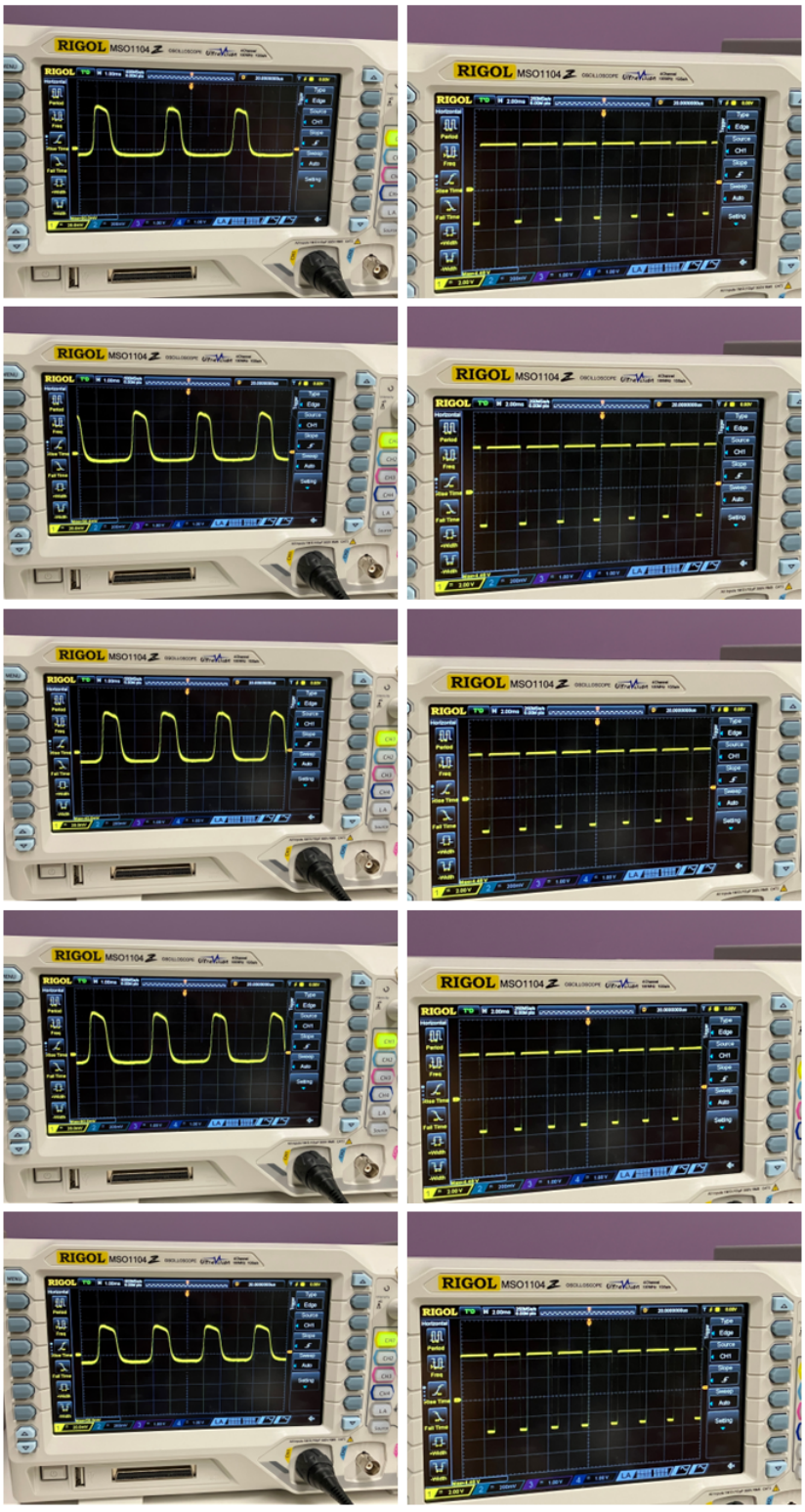

Figure 18: 5 lowest frequency notes filtered (left) and unfiltered (right)

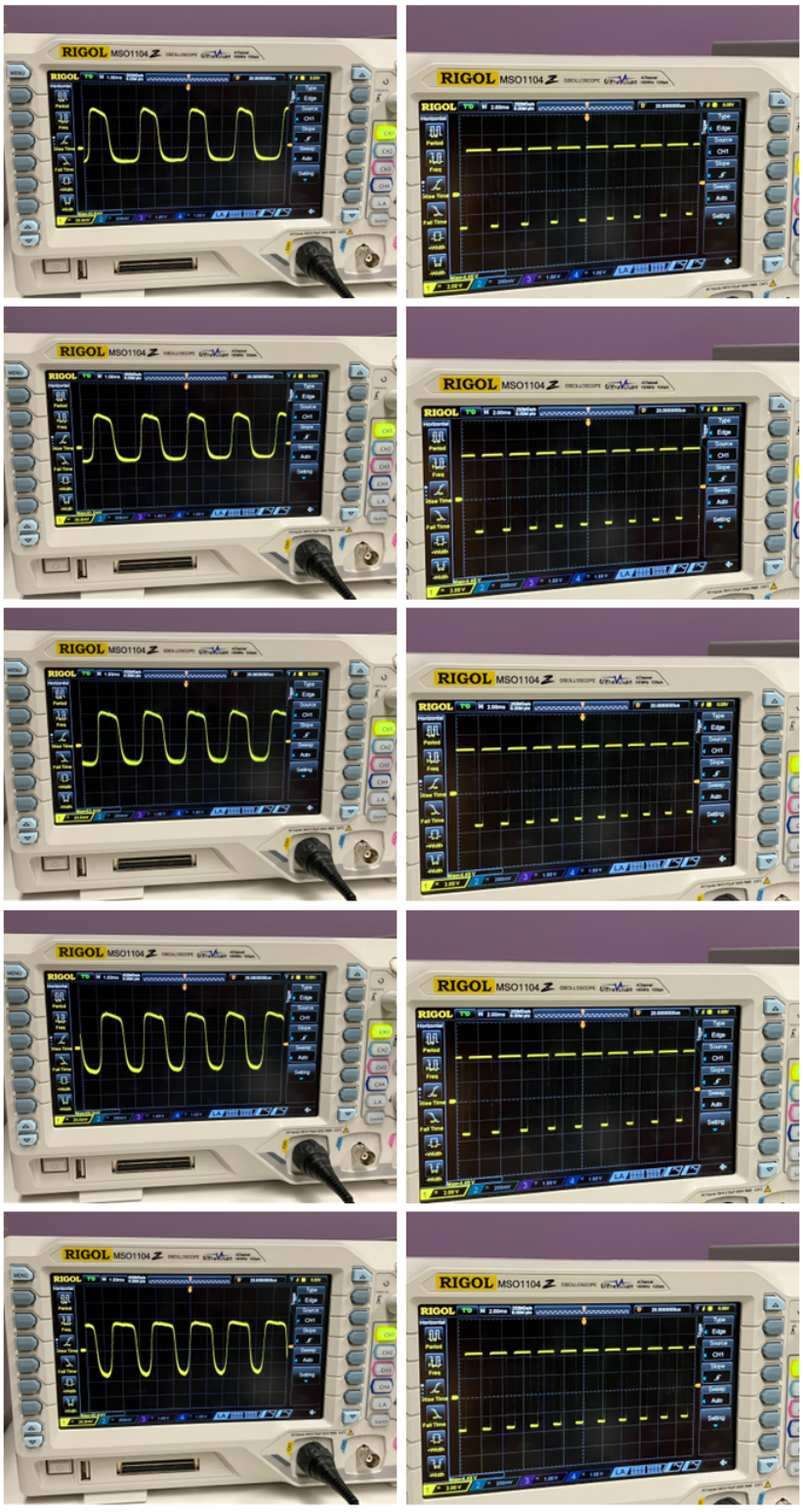

Figure 19: 5 middle frequency notes filtered (left) and unfiltered (right)

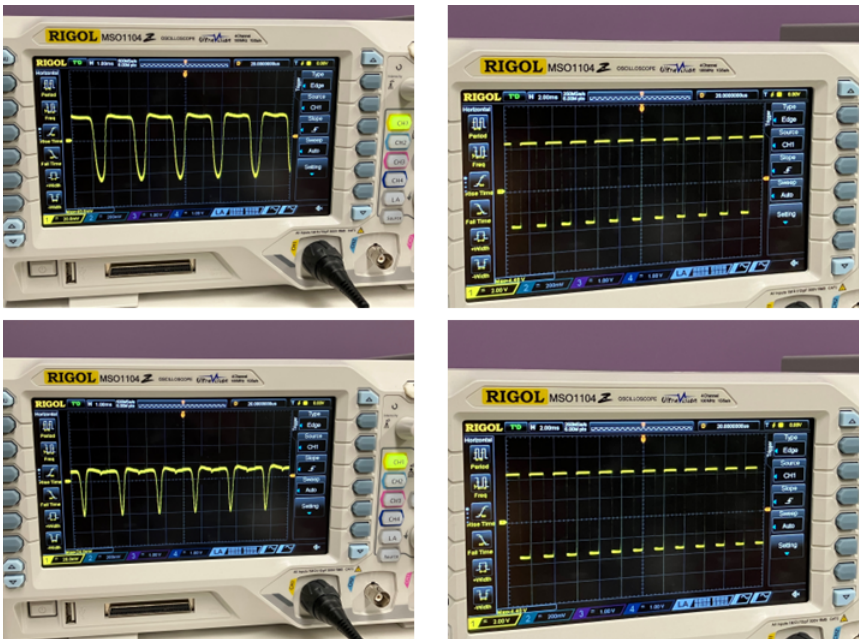

Figure 20: 2 highest frequency notes filtered (left) and unfiltered (right)

Note	Desired Frequency (Hz)	Measure Frequency (Hz)	Error (Hz)	Error (%)
C	261.63	262.30	0.67	0.256
C#	277.18	277.40	0.22	0.079
D	293.66	292.00	-1.66	-0.565
D#	311.13	310.00	-1.13	-0.363
E	329.63	330.90	1.27	0.385
F	349.23	349.80	0.57	0.163
F#	369.99	368.80	-1.19	-0.322
G	392.00	392.70	0.70	0.179
G#	415.30	415.10	-0.20	-0.048
A	440.00	439.20	-0.80	-0.182
A#	466.16	466.40	0.24	0.051
B	493.88	495.00	1.12	0.227

Table 4: Comparison of theoretical frequencies for each note with the actual measured frequency.

6.3 Volume Controller (VC)

The components values were the volume controller were determined experimentally and the following results were obtained.

Figure 21: High amplitude waveform when Arduino PWM to analog converter outputs 0.3 V while the A note is being played.

Figure 22: Low amplitude waveform when Arduino PWM to analog converter outputs 0.1 V while the A note is being played.

This shows that the amplitude of the resultant waveform can be controlled through setting the voltage of an analog pin of the Arduino (i.e., changing the duty cycle of the PWM signal). The volume controller circuit

devised successfully worked.

6.4 Software Interface

The computer communicates with the Arduino via USB. When a key is pressed, an ASCII character is serially sent to Arduino to represent a corresponding action to take. Keys a,w,s,e,d,f,t,g,y,h,u,j play different notes as shown in Figure 23. Pressing 'm' switches the mode between piano and synthesizer mode. In piano mode, notes fade out upon being pressed while synthesizer mode sustains the note until released. The up and down arrow keys can also be pressed to adjust the volume between 1 and 5.

Figure 23: GUI run on a computer connected to the Arduino

7 Conclusion

The development of this electric piano was a success. Digital signals were converted into analog voltages with minimal variance from expected values. The waveforms generated had accurate frequencies, allowing the piano to play all 12 notes of the chromatic scale. A low pass filter cleaned up harsh sounding harmonics and provided a smooth sound to the output. An additional volume decay allowed the user to control the amplitude of the waveform without loss of sound quality. A GUI also provided a user-friendly experience to the user playing the piano. All of these features were implemented with high degrees of success and much knowledge about circuit operation, assembly, and implementation was gained.